Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise

ans:

```
#!/bin/bash
if [ -f "myfile.txt" ];
then
echo "File exist"
else
echo "$name not found"
fi
```

===

Assignment 2: Write a script that reads numbers from the user until they enter ’0’. The script should also print whether each number is odd or even.

ans:

```
#!/bin/bash
while true; do
echo "Enter a number:"
read number
if [ $number -eq 0 ]; then
echo "Exiting"
break
fi
if [ $((number % 2)) -eq 0 ]; then
echo "$number is even"
else
echo "$number is odd"
fi
Done
```

==

Assignment 3: Create a function that takes a filename as an argument and prints the number of lines in the file. Call this function from

ans:

```
#!/bin/bash
countLines(){
local filename="$1"
local lines=$(wc -l <"$filename")
echo "Number of lines in $filename : $lines"
}
countLines "myfile.txt"
countLines "second.sh"
countLines "third.sh"
```

===

Assignment 4: Write a script that creates a directory named TestDir and inside it, creates ten files named File1.txt, File2.txt, ... File10.txt. Each file should contain its filename as its content (e.g., File1.txt contains "File1.txt

ans:

```
#!/bin/bash
mkdir TestDir
cd TestDir
for((i=1;i<=10;i++))
do
name="File${i}"
```

```
echo "$name">"$name"
done
```

===

Assignment 5: Modify the script to handle errors, such as the directory already existing or lacking permissions to create files. Add a debugging mode that prints additional information when enabled.
ans:

```
#!/bin/bash
if [ "$DEBUG" = "true" ]; then
set -x
fi
errorHandler(){
echo "Error:$1"
exit 1
}
if [ -d "TestDir" ]; then
errorHandler "Directory Already Exists"
fi
mkdir -p TestDir || errorHandler "Failed to create Directory"
cs TestDir || errorHandler "Failed to change Directory"
for(( i=1;i<=10;i++ )); do
echo "File$i.txt">"File$1.txt"||errorHandler "Failed to
create ’File$i.txt’"
done
if [ "$DEBUG"="true" ]; then
set+x
fi
```

===

ssignment 6: Given a sample log file, write a script using grep to extract all lines containing "ERROR". Use awk to print the date, time, and error message of each extracted line. Data
ans:

```
#!/bin/bash
log_file="sample_log.txt"


grep "ERROR" "$log_file" | awk ’{print $1, $2, $NF}’ | awk ’{print $1" "$2": "$3}’
```